НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

До захисту допущено:
Завідувач кафедри
_____ Сергій СТІРЕНКО
«___»_____20__ р.

Дипломний проект

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Комп'ютерні системи та мережі»

спеціальності 123 «Комп'ютерна інженерія»

на тему: «Система забезпечення політики безпеки доступу до документів»

Виконав:
студент IV курсу, групи ІО-61
Літвін Олександр Юрійович _____

Керівник:
Доцент, кандидат технічних наук
Верба Олександр Андрійович _____

Консультант з нормоконтролю:
Професор, доктор технічних наук
Сімоненко Валерій Павлович _____

Рецензент:
Асистент
Алещенко Олексій Вадимович _____

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 123 «Комп'ютерна інженерія»
Освітньо-професійна програма «Комп'ютерні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Сергій СТІРЕНКО
«___»_____20__ р.

ЗАВДАННЯ
на дипломний проект студенту
Літвіну Олександру Юрійовичу

1. Тема проекту «Система забезпечення політики безпеки доступу до документів», керівник проекту Верба Олександр Андрійович, доцент, кандидат технічних наук, затверджені наказом по університету від «07» травня 2020 р. № 1081-с
2. Термін подання студентом проекту 13 червня 2020 р.
3. Вихідні дані до проекту: технічне завдання, науково-технічна література
4. Зміст пояснювальної записки: порівняльний аналіз існуючих програмних рішень, вибір засобів реалізації та опис отриманої системи

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо) :

1. Функціональна схема – плакат
2. Принципова схема – плакат
3. Структурна схема – плакат

6. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Сімоненко В.П., професор		

7. Дата видачі завдання 01 вересня 2019 р.

Календарний план

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітка
1	Затвердження теми роботи	01.09.2019-22.12.2019	
2	Вивчення та аналіз завдання	23.12.2019-20.03.2020	
3	Розробка архітектури та загальної структури системи	20.03.2020-01.04.2020	
4	Розробка структур окремих підсистем	01.04.2020-10.04.2020	
5	Програмна реалізація системи	11.04.2020-20.04.2020	
6	Оформлення пояснювальної записки	01.05.2020-23.05.2020	
7	Передзахист	24.05.2020-26.05.2020	
8	Захист	15.06.2020-20.06.2020	

Студент Олександр ЛІТВІН

Керівник Олександр ВЕРБА

АНОТАЦІЯ

Дана дипломна робота присвячена розробці системи забезпечення політики безпеки доступу до документів з метою застосування для спостереження та виявлення витоків конфіденційної інформації.

У роботі був проведений розгляд предмету інформаційної безпеки, аналіз існуючих прикладів систем протидії витоку інформації, технологій, що в них використовуються. На основі поточного стану технологій забезпечення безпеки інформації було розроблено систему з використанням файлів-агентів і режимів доступу до них.

Програмний продукт реалізовано на мові програмування Python з використанням фреймворку Flask. Інтерфейс користувача має вигляд консольної програми.

ANNOTATION

This thesis is devoted to the development of a security policy management system for document access in order to use to monitor and detect leaks of confidential information.

The subject of information security was considered in the project. Examples of existing systems for counteracting leakage of information and technologies used in them are analyzed. Based on the current state of information security technologies, a system was developed using agent files and modes of access to them.

The software product is implemented in the Python programming language using the Flask framework. The user interface looks like a console program.

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЕКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проект	2	
2	А4	*ІАЛЦ. 467800.001* ВП	Відомість проекту	1	
3	А4	*ІАЛЦ. 467800.002* ТЗ	Технічне завдання	4	
4	А4	*ІАЛЦ. 467800.003* ПЗ	Пояснювальна записка	54	
5	А4	*ІАЛЦ. 467800.004* Д1	Функціональна схема	1	
6	А4	*ІАЛЦ. 467800.005* Д2	Принципова схема	1	
7	А4	*ІАЛЦ. 467800.006* Д3	Структурна схема	1	
8	А4	*ІАЛЦ. 467800.007* Д4	Текст програми	6	

				ІАЛЦ. 467800.001 ВП		
	ПІБ	Підп.	Дата			
Розробн.	Літвін О.Ю.			Відомість дипломного проекту	Лист	Листів
Керівн.	Верба О.А.				1	1
Консульт.					КПІ ім. Ігоря Сікорського Каф. ОТ Гр. ІО-61	
Н/контр.	Сімоненко В.П.					
Зав.каф.	Стіренко С.Г.					

Технічне завдання

до дипломного проекту

на тему: «Система забезпечення політики безпеки доступу до документів»

Київ – 2020 року

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Технічне завдання описує розробку програмного рішення за темою «Система забезпечення політики безпеки доступу до документів». Область застосування даного рішення є спостереження і виявлення можливих каналів витоку інформації.

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки рішення є завдання на виконання роботи кваліфікаційно-освітнього рівня «бакалавр комп'ютерної інженерії», затверджене кафедрою обчислювальної техніки Національного технічного Університету України «Київський Політехнічний інститут ім. Ігоря Сікорського».

3. МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою дипломного проекту є розгляд предмету інформаційної безпеки і розробка системи для забезпечення політики безпеки доступу до документів.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелами розробки є науково-технічна література з інформаційних технологій та інформаційної безпеки, технічна документація до програмного забезпечення, утиліт та бібліотек, використаних під час розробки, публікації в періодичних виданнях та ресурси мережі Інтернет щодо даного питання.

5. ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до програмного продукту, що розробляється

- Генерація файлів-агентів.
- Інтерфейс користувача має бути простим і зрозумілим для застосунку.
- Можливість робити аналіз зібраних сервером даних.
- Редагування режимів доступу до файлів

5.2. Вимоги до програмного забезпечення

- Мова програмування Python.
- Використання бібліотеки Flask.

5.3. Вимоги до апаратного забезпечення

- Комп’ютер на базі процесору Intel Pentium 4 / Athlon 64 і вище.
- Оперативної пам’яті не менше 1024 Мбайт.
- 1 Гбайт вільного місця на пристрої зберігання інформації.
- Підключення до мережі Інтернет з виділеною адресою або доменним ім’ям.

6. Етапи розробки

	Дата
Вивчення літератури	20.12.2019
Складання та узгодження технічного завдання	15.01.2020
Проектування програмного забезпечення	27.01.2020
Програмна реалізація продукту	14.02.2020
Тестування програмного комплексу	01.05.2020
Отладка і виправлення помилок	15.05.2020
Оформленняе документації дипломної работи	06.06.2020

Пояснювальна записка

до дипломного проекту

на тему: «Система забезпечення політики безпеки доступу до документів»

Київ – 2020 року

ЗМІСТ

ПЕРЕЛІК ТЕРМІНІВ ТА СКОРОЧЕНЬ

ISO	(англ. International Organization for Standardization) — міжнародна організація, метою діяльності якої є ратифікація розроблених спільними зусиллями делегатів від різних країн стандартів.
FTP	(англ. File Transfer Protocol) — протокол передачі файлів.
P2P	(англ. Peer-to-peer) —в архітектури системи, в основі якої стоїть мережа рівноправних вузлів.
IPC	(англ. Information Protection and Control) — технологія захисту конфіденційної інформації від внутрішніх загроз.
DLP	(англ. Data Leak Prevention) — технології запобігання витоків конфіденційної інформації з інформаційної системи зовні, а також технічні пристрої (програмні або програмно-апаратні) для такого запобігання витоків.
IM	(англ. Instant messaging) — миттєві повідомлення.

USB	(англ. Universal Serial Bus) — універсальна Послідовна Шина.
WiFi	(англ. Wireless Fidelity) — бездротова правдивість відтворення.
LPT	(англ. Line Print Terminal) — паралельний порт.
COM	— загальний домен верхнього рівня.
HTTP	(англ. Hypertext Transfer Protocol) — протокол передачі гіпертекстових документів.
HTTPS	(англ. HyperText Transfer Protocol Secure) — розширення протоколу HTTP для підтримки шифрування з метою підвищення безпеки.
SMTP	(англ. Simple Mail Transfer Protocol) — простий протокол передачі пошти.
SAN	(англ. Storage Area Network) — мережа зберігання даних.
NAS	(англ. Network Attached Storage) — сервер для зберігання даних на файловому рівні.
IDS	(англ. Intrusion Detection System) — система виявлення вторгнень.

UI	(англ. User interface) — користувацький інтерфейс.
TCP	(англ. Transmission Control Protocol) Протоко́л керування передачею.
HTML	(англ. Hyper Text Markup Language) — мова текстової розмітки
IP	(англ. Internet Protocol) Інтернет протокол
YAML	(англ. Yet Another Markup Language) — мова опису розмітки

ВСТУП

На сьогодні комп'ютерна злочинність складається не лише з типових комп'ютерних злочинів, перспективні комп'ютерні технології не тільки відкрили нові горизонти для бізнес-сфери, науки, освіти та політики, але і створили сприятливі умови для появи нових видів злочинності. Головним фактором, що впливає на кількість скоєних злочинів в конкретно взятій країні чи світі, на думку багатьох авторів, є ступінь проникнення комп'ютерних технологій у всі сфери людського життя. І справді останнім часом країни світу все частіше стикаються з проблемами регулювання правових відносин у сфері комп'ютерних технологій. Було б набагато простіше, якби просто попередити подібні ситуації. Поки законодавчі органи створюють та удосконалюють правову базу щодо технологічного покращення умов життя людини та караючі заходи для кіберзлочинців, виробники технічного та програмного забезпечення намагаються вирішити цю проблему своїми методами.

Об'єктом дипломної роботи є система забезпечення політики безпеки доступу до документів. Вибір об'єкту зумовлений моїм індивідуальним інтересом до цієї теми, а також стрімким ростом популярності теми захисту інформації.

Предметом дослідження є створення системи спостереження з використанням файлів-агентів.

Метою дипломної роботи є:

- Вивчення предмету інформаційної безпеки, методів захисту інформації, аналіз існуючих рішень.
- Розгляд доцільності та практичності використання знайдених методів, перегляд можливих варіантів реалізації системи.
- Виявлення можливих проблем враховуючи обрані технології.

Були поставлені наступні завдання:

- Перегляд історичного досвіду розробки систем захисту інформації
- Аналіз нинішнього стану технологій попередження, виявлення та спостереження витоків інформації.
- Реалізація системи для спостереження і виявлення витоків інформації, що вже відбулися.

Дипломна робота складається зі вступу, трьох розділів та висновків до проекту. У першому розділі розглянуто предмет інформаційної безпеки та аналіз програмних рішень захисту інформації. У другому розділі описуються технології використані у існуючих системах та у створенні проектного продукту. У третьому розділі описується реалізація всіх складових системи, функціонал, принцип роботи. У висновках проекту підбиваються підсумки та робляться остаточні заключення щодо теми проекту.

РОЗДІЛ 1
ІНФОРМАЦІЙНА БЕЗПЕКА, МЕТОДИ ЗАХИСТУ ІНФОРМАЦІЇ.

Предмет інформаційної безпеки.

Продовжує тривати час стрімкого розвитку технологій та переорієнтація сучасного бізнесу у простір технології, орієнтири компаній зсуваються в сторону нематеріальних активів, серед яких інформація впевнено займає місце. Кожна компанія індивідуально визначає, які дані є важливими та які мають характер конфіденційності.

Конфіденційна інформація — це інформація про фізичну особу, а також інформація, доступ до якої обмежено фізичною або юридичною особою, крім суб'єктів владних повноважень. Доступ до такої інформації та її поширення можливі лише за згодою її власників (тобто тих, кого ця інформація безпосередньо стосується) та на тих умовах, які вони вкажуть. Відповідно до Ст. 21 ЗУ «Про інформацію» конфіденційна інформація разом із службовою та таємною інформацією належить до інформації з обмеженим доступом [1].

Всесвітня мережа та нові технології проникають все тісніше у повсякденне життя людини. Це сприяє збільшенню онлайн користувачів, гострої необхідності у користуванні цифровими інфраструктурами, кібер-злочинності та зниженням рівня культури безпеки. За останні 7 років в Україні кількість інформаційних злочинів, до яких входять спрямовані на заволодіння інформацією (для користування у власних цілях або для продажу зацікавленим особам), зросла щонайменше у 2,5 рази. Це пов’язано не лише зі збільшенням спроб вчинити інформаційні злочини, а також через збільшення фахівців, здатних ці злочини виявити, та відсутність фахівців правової сфери, здатних гідно оцінити ступінь злочину та покарання за його

вчинення. Виявлення в більшій степені можливе завдяки посиленню інформаційної безпеки на всіх рівнях організації інформаційних технологій.

Інформаційна безпека є комплексом, що складається з системи дій, стандартів, інструкцій та програмного забезпечення для попередження несанкціонованого доступу чи використання, запису та знищення інформації а також для розкриття інцедентів витоку інформації чи аналізу стану інфраструктури. Основним завданням інформаційної безпеки є баланс між захистом інформації: конфіденційності, цілісності і доступності даних та доцільністю застосування мір захисту, обов'язково без втрати продуктивності компанії чи організації. Цей баланс досягається багатоетапним процесом управління ризиками, забезпеченням ідентифікації основних засобів та нематеріальних активів, джерел загроз, вразливих ланок, можливості управління ризиками [2]. Основні принципи захисту інформації в комп'ютерних системах вперше виділили Джеррі Зальцер і Майкл Шрьодер в 1975 році в статті «Захист інформації в комп'ютерних системах». Згодом категорії стандартизували та визначили за наступними найменуваннями[14]:

- Конфіденційність — властивість інформації бути конфеденційною (незагальною або закритою) для неавторизованих користувачів, сутностей чи процесів;
- Цілісність — властивість зберігати правильність і повноту активів;
- Доступність — властивість інформації бути доступною і готовою до використання за запитом авторизованого суб'єкта, що має на це право.

Багато дослідників у цій сфері та різні організації намагалися удосконалити або запропонувати свої моделі інформаційної безпеки, такі як Паркерівська гексада Дона Паркера, «Три основоположних принципи комп'ютерної безпеки» Міністерства оборони США, стандарт управління інформаційною безпекою O-ISM3 міжнародного консорціуму The Open

Group та інші. З усіх вище перечисленних моделей інформаційної безпеки класична тріада CIA досі залишається найбільш визнаною і поширеною у міжнародному професійному співтоваристві. Хоча не кожен з них згоден, що тріада CIA повністю відповідає технологіям і потребам бізнесу, що останнім часом набирають швидких обертів. Результатом дискусій довкола цієї теми стали рекомендації про необхідність прирівняти безпеку до недоторканності приватного життя, а також затвердити додаткові принципи. Наступний перелік був включений в стандарти Міжнародної організації по стандартизації (ISO):

- автентичність — властивість, що гарантує, що суб'єкт або ресурс ідентичні заявленим;
- підзвітність —суб'єкт несе відповідальність за свої дії і рішення;
- неможливість відмови — здатність засвідчувати можливість існування події або дії і їх суб'єктів так, щоб ці події або дії і суб'єкти, які мають до них відношення, не могли бути поставлені під сумнів;
- достовірність — властивість відповідності передбаченому поведінки і результатів.

Інформаційна безпека організації(компанії чи підприємтва) стан захищеності корпоративних даних, у якому забезпечена їх конфіденційність, цілісність, автентичність і доступність, тобто принципи, що затверджені стандартами для забезпечення інформаційною безпеки. Або це перелік відповідних заходів, направлених на забезпечення захисту даних. Задачею системи інформаційної безпеки організації[14]:

- забезпечення захищеного зберігання інформації на носіях;
- захист даних, що передаються по каналах зв'язку;
- створення резервних копій, післяаварійного відновлення і т. д.

Компанія має сприяти інформаційній безпеці та швидко відновлюватися після будь-який інцендентів не лише з мінімальною втратою продуктивності а й не зупиняючи постачу послуг, тобто виконуючи свою основну роботу. Увімкнені Політики інформаційної безпеки допомагають забезпечити безпечну IT-середовище для забезпечення потреб клієнтів компанії, стабільності та неперервності бізнес-активів IT. Для допомоги організаціям у підготовці Керівництво зі стандартів операційними процесами (SOP) для будь-якої галузі та сфери діяльності. Може бути налаштовано відповідно до потреб і операцій політик безпеки інформаційної компанії. Інформаційна безпека організації досягається лише повним комплексом відповідних організаційні та технічних заходів, спрямованих на захист корпоративних даних. Для початку переходу у стан інформаційної безпеки необхідно виконати п'ять ключових етапів:

- оцінка вартості;
- розробка політики безпеки;
- реалізація політики;
- кваліфікаційна підготовка спеціалістів;
- аудит.

Схему, що зображує можливі канали витоку інформації можна переглянути на Рис. 1.1., що наведено нижче.

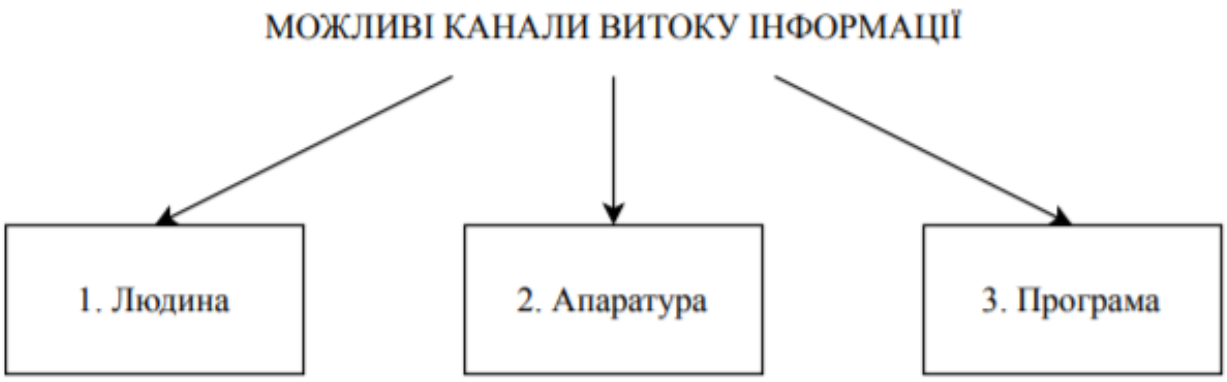


Рис. 1.1. Джерела витоку інформації.

В даний час однією з найбільш актуальних загроз у сфері інформаційної безпеки є витік конфіденційних даних від несанкціонованих дій користувачів. Втрата інформації припускає незаконний перехід конфіденційних відомостей до особи, що не має права використовувати ці відомості у своїх цілях для одержання прибутку або передачі іншій особі. У тому випадку, коли втрата інформації відбувається з вини персоналу — втрата інформації позначається терміном розголошення або розголос інформації. Розголошення інформації завжди здійснюється людиною усно, письмово, за допомогою жестів, міміки, умовних сигналів. Термін "витік інформації" більшою мірою стосується втрати інформації за рахунок її перехоплення за допомогою технічних засобів розвідки.

Особливістю учасника, якого називають зловмисник є цілеспрямованість, тобто навмисні, усвідомлені спроби одержання конкретної інформації, навмисний і таємний пошук чи формування каналів викрадення інформації. Канали втрати конфіденційної інформації поділяють на організаційні та технічні. Організаційні канали розголошення інформації, утворені на основі різноманітних (не виключено, що законних) взаємин з організацією або співробітниками цієї організації для подальшого несанкціонованого доступу до конфіденційної інформації. Далі розглянуто

основні способи утворення організаційних каналів для подальшого використання зловмисником:

- влаштування зловмисником у якості співробітника у організацію, є дуже розповсюдженим підходом, найбільш вірогідними посадами для цього є технічні, допоміжні або другорядні посади;
- установлення зловмисником довірчих взаємин зі співробітником фірми або особами, що мають право вільного доступу в даній фірмі;
- отримання кримінального, силового доступу до інформації, як викрадення документів, справ, цифрових накопичувачів чи комп'ютерів; шантаж, підкуп окремих працівників; інсценування екстремальних ситуацій;
- отримання інформації з випадкового каналу, що спричинена витоком інформації.

Інсайдер — робітник компанії, який має доступ до конфіденційної інформації, розміщеної у комп'ютерній мережі установи. Внутрішні порушники поділяються на наступні категорії: лояльні інсайдери (недбалі та маніпульовані); скривджені та нелояльні інсайдери; мотивовані ззовні (мотивовані фінансово та впроваджені); інші порушники (ті, що мають на меті вплинути на вартість акцій підприємства) [3].

Протидія інсайдерству має здійснюватися безперервно, адже кожен співробітник, який має доступ до інформації є потенційним порушником. При цьому необхідно дотримуватись балансу між закритістю інформації та її доступністю для працівників компанії, інакше вони не зможуть виконувати свої прямі обов’язки. Основні напрямки захисту від інсайдерів: захист документів; захист каналів витоку; моніторинг дій користувачів [4].

Огляд існуючих рішень для захисту інформації.

Наступним показом за статистикою для аналізу витоків інформації складають витоки, що пов'язані з IT-інфраструктурою організації, тобто внутрішні загрози. Під внутрішньою загрозою розуміються навмисні дії, як інсайд, коли один із співробітників свідомо виносить корпоративний секрет назовні. Великий відсоток і немалу небезпеку складають також ненавмисні внутрішні інциденти.

Для захисту від ненавмисних витоків інформації спеціалізовані продукти набули високого попиту, як IPC — технологію захисту конфіденційної інформації від внутрішніх загроз. Рішення класу IPC призначені для захисту інформації від внутрішніх загроз, запобігання різних видів витоків інформації, корпоративного шпигунства і бізнес-розвідки[5].

IPC-система є поєднанням двох основних і важливих технологій:

- шифрування носіїв інформації на всіх етапах у мережі;
- використання технології DLP для контролю технічних каналів витоку інформації.

Технологія IPC є логічним продовженням технології DLP, що дозволяє захищати дані не тільки від витоку технічними каналами, а й від несанкціонованого доступу користувачів до інфраструктури, як мережа, інформація, додатки організаці. А значним стрибком став можливий захист, у випадку коли пристрої потрапляють до рук зловмисників, що означає захист від витоку при нелегальному доступі третьою людиною до безпосереднього носія інформації.

Завданням технології IPC-систем це запобігти передачі конфіденційної інформації за межі корпоративної інформаційного простору. Така передача (витік) може бути навмисною або ненавмисною. Ознайомившись зі статистикою визначаємо що більша частина витоків відбувається випадково,

через помилки, неуважність та недбалість, працівників – виявляти подібні випадки набагато простіше. Інша частина пов'язана зі спеціально направленим комплексом дій на викрадення, наприклад операторів чи користувачів інформаційних систем підприємства, зокрема, промисловим шпигунством, конкурентної розвідкою.

Додаткові завдання систем класу IPC:

- система має запобігати передачі назовні конфіденційної, а й із зовні небажаної інформації як спам, зайві обсяги даних та інше;
- система має запобігати передачі небажаної інформації не тільки зсередини назовні, а й із зовні всередину, для захисту інфрастракутури інформаційної системи організації;
- система має запобігати використання працівниками Інтернет-ресурсів і ресурсів мережі в особистих цілях;
- система має захищати від спаму;
- система має захищати від вірусів;
- система має забезпечувати оптимізацію завантаженних каналів, зменшення нецільового трафіку;
- облік робочого часу і присутності на робочому місці;
- система має забезпечувати відстеження благонадійності співробітників, оскільки основний відсоток витоку конфеденційної інформації трапляється через співробітників;
- система має забезпечувати резервні копії (архівацію) інформації для забезпечення додаткового захисту;
- система має захищати від випадкового або навмисного порушення внутрішніх нормативів;
- забезпечувати відповідності стандартів у галузі інформаційної безпеки і чинного законодавства.

Враховуючи вище наведену інформацію, виділимо рішення класу DLP, що застосовується в IPC для підтримки контролю технічних каналів, через які потенційно може відбутися витік конфіденційної інформації:

- електронна пошта;
- веб-додаток пошти;
- соціальні мережі й блоги;
- мережі файлообміну;
- форуми та інші інтернет-ресурси, у тому числі виконані на AJAX-технології;
- IM (WhatsApp, Skype, WeChat, Google Talks, Line, Facebook Messenger);
- P2P-клієнти;
- периферійні пристрої (USB, LPT, COM, WiFi, Bluetooth й інше);
- принтери локальних та мережевих систем.

Технології DLP в IPC підтримують контроль над протоколами обміну даними:

- HTTP;
- HTTPS (SSL);
- FTP;
- FTP-over-HTTP;
- FTPS;
- SMTP.

IPC має обов'язковим компонентом архів, що дозволяє відслідковувати інформацію щодо обраних потоків даних (пакетів, повідомлень). Тобто він забезпечує зберігання інформації про дії співробітників у одній чи декількох пов'язаних між собою баз даних. IPC-системи є лідерами у сфері

забезпечення безпеки, такі системи дозволяють архівування всіх каналів витоку, які контролюються ними. У такому архіві системи IPC збережено копії вивантажених у Інтернет документів і текстів електронних листів, роздрукованих документів і файлів, записаних на периферійні пристрої. Архів забезпечує доступ адміністратору ІБ у будь-який момент часу для огляду, аналізу та перевірки будь-якого файла чи документа в архіві або скориставшись лінгвістичним пошуком інформації за єдиним архівом (або всіма розподіленими архівами водночас) тексту в архіві. Усі повідомлення, збережені у архіві за необхідності можна переглянути або переслати, а будь-який завантажений в Інтернет, записаний або роздрукований на зовнішній пристрій файл або документ, переглянути або скопіювати. Це дозволяє зробити ретроспективний аналіз можливих витоків, а також проаналізувати систему, зробити комплекс дій направлених на підвищення безпеки організації, її інформаційної інфраструктури чи попередити витік конфеденційної інформації.

Технології IPC можуть забезпечувати шифрування інформації у ключових точках мережі, які наведено далі:

- ноутбуки;
- персональні комп’ютери;
- жорсткі диски серверів;
- SAN;
- NAS;
- магнітні стрічки;
- зовнішні пристрої.

Технології класу IPC використовують криптографічні модулі, у тому числі DES, Triple DES, RC5, RC6, AES, XTS-AES, що є найбільш ефективними алгоритмами.

IPC-системи мають агенти в усіх ключових точках мережі: сервери, сховища, шлюзи, ПК/ноутбуки, периферійні та мережеві пристрої користувача. Технології IPC реалізовані для Windows, Linux, Sun Solaris, Novell.

Термін набув великого спектру значень. Так DLP-системою можна назвати навіть антивірусну програму, оскільки вона є запобіжним заходом та попереджує встановлення троянів, що надсилають інформацію з локального пристрою до вказоного місця його творцем. Чи програму блокатор USB-портів, адже вона запобігає витокаму інформації через USB-накопичувачі. Проте обидві системи згадані вище не можуть та не захищають організацію від витоків – вони лише блокують один з каналів, усуваючи одну з причин. Для деяких організації «латання» вразливих частин системи може бути тимчасовим рішенням, проте для великих організацій це буде неприйнятно. DLP-системою називають комплексне рішення корпоративного масштабу, що запобігає та захищає від витоків інформації з різних каналів і причин. Таке визначення є лише частково коректним, оскільки гіпотетична система, що фізично блокує всі порти і перекриває доступ до Інтернет є DLP-системою згідно визначення, але зовсімне є нею згідно користувацьких вимог. [6]

Для забезпечення потреб функціонування та захисту сучасні організації мають вузький, але вкрай необхідний список вимог: Інтернет та доступ до мобільних накопичувачів. Це означає, що перекриття цих каналів унеможливлює роботу, що є великою проблемою, а отже доступ до них слід контролювати. Враховуючи вище згадане, необхідно наголосити на особливому функціоналі DLP-систем, що вирізняє її від більшості рішень у просторі інформаційної безпеки. DLP-система аналізує трафік, який надходить каналами та у випадках коли трафік виявився секретним «голосно кричати» адміністратору.

При розгляді DLP-систему в аспекті одного каналу, її можна порівняти з «чорним ящиком»: йому на вхід подається інформація, що прямує каналом, а на виході він формує заключення щодо секретності вхідної інформації. Розглянутий принцип не залежить від специфіки каналу, може бути використана в алгоритмі винесення заключення.

Таким чином, основна цінність DLP-системи полягає в алгоритмі, на основі якого працює «чорний ящик». Архітектура і навіть список підтримуваних каналів є вторинними характеристиками даного ПЗ. Проте, з погляду кінцевого замовника, дані чинники також дуже важливі, оскільки без них додаток практично марний

За даними компанії Perimetrix ефективність фільтрації із застосуванням контентних технологій досягає лише 80 %. Така статистика означає, що 20 % корпоративних секретів можуть безперешкодно покинути мереду компанії. Крім того, система фільтрації контента допускає помилки другого типу, визнаючи деякі легітимні відомості секретними. [15]

Оскільки втрата даних стала серйозною проблемою, що може нести не лише збитки напряму, а і збитки через втрати репутації та довіри до компанії. Зі поширенням відомостей, рівня організації безпеки та застосування більш складних заходів забезпечення безпеки зростають і кількість інцендентів з витоку інформації. Так наприклад у 2009 році було зареєстровано 735 інцендентів з втрати даних, що на 39% більше, ніж зареєстрованих інцендентів за 2008 (Open Security Foundation, 2009). При середній втраті записів кількістю один інцидент, що перевищує 750 000, витрати організації, фінансові втрати, штрафи та репутаційні, а також судові виплати можуть складати мільйони доларів.

Організації мають бути готові до захисту у різних напрямах, оскільки як згадано раніше інцендент може статися з середини або із зовні; чи випадково як результат порушення безпеки працівника. З розвитком безпеки, розвивається різноманіття причин втрати даних, оскільки популярні медіа-

ресурси, як соціальні мережі чи ІМ системи, надали нові канали для втрати даних, у той же час гнучкість роботи призвело до росту числа портативних пристроїв, що здатні зберігати великі об’єми інформації та забезпечувати доступ. DLP стала вирішенням деяких проблем, технологія розвивалася з плином часу разом з прогресом інформаційних злочинів. Задачею останніх рішень DLP став захист важливих даних, конфединційних даних організації, де б вони не знаходились, шляхом їх виявлення у стані спокою (у сховищі), при використані чи при передачі у мережі. Незважаючи, що рішення DLP можуть бути інструментом запобігання інцидентів, з втратою даних, та допомоги організації, імплементувати цю технологію згідно до нормативних вимог, законодавства доволі складно за словами Gartner. Comsec розробила перевірені методики для забезпечення допомоги організаціям у розробці стратегій DLP, заснованих на найкращих практикаї та міжнародних стандартах. Від оцінки ризиків існуючої середи безпеки, складання карт уснуючих систем DLP, документування ролей DLP, забов’язань і процесів, та програм навчання і сповіщення співробітників; Comsec може надати комплексну консультацію, для допомоги організації успішно розгорнути рішення DLP. На Рис. 1.2. можна ознайомитись з діаграмою витоку даних.

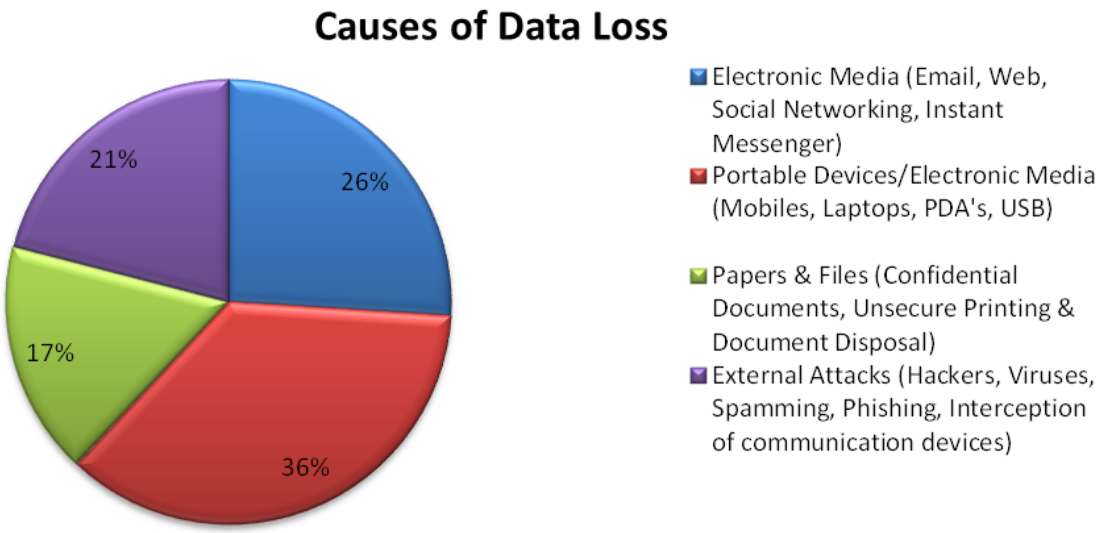


Рис. 1.2. Діаграма втрат даних[16]

Практично половина сучасних витоків відбувається в результаті крадіжки мобільних пристроїв. Єдиним способом захисту тоді є шифрування – адже фізичний захист, паролі на даному етапі розвитку обчислювальних систем стали не актуальними методами захисту. Проблемою DLP полягає лише в тому, що переважна більшість рішень класу DLP не може забезпечити шифрування, а рішення по шифруванню не можуть здійснювати фільтрацію витікаючого трафіку.

Підприємствам і організаціям необхідний уніфікований функціонал для досягнення таких цілей:

- отримання однієї повноцінної системи захисту від усіх загроз витоку інформації, а не декілька окремих;
- використовувати єдині інтегровані політики для фільтрації і шифрування документів;
- забезпечити просту і швидку відповідність різним нормативним актам і стандартам.

InfoWatch Traffic Monitor

Ця система э розподіленою і багатокомпонентною системою, що призначена для контролю трафіквихідних листів, переданих за протоколами SMTP і HTTP.

Архітектуру такої системи, номер версії 6.7 можна переглянути на Рис. 1.2. що наведено нижче.

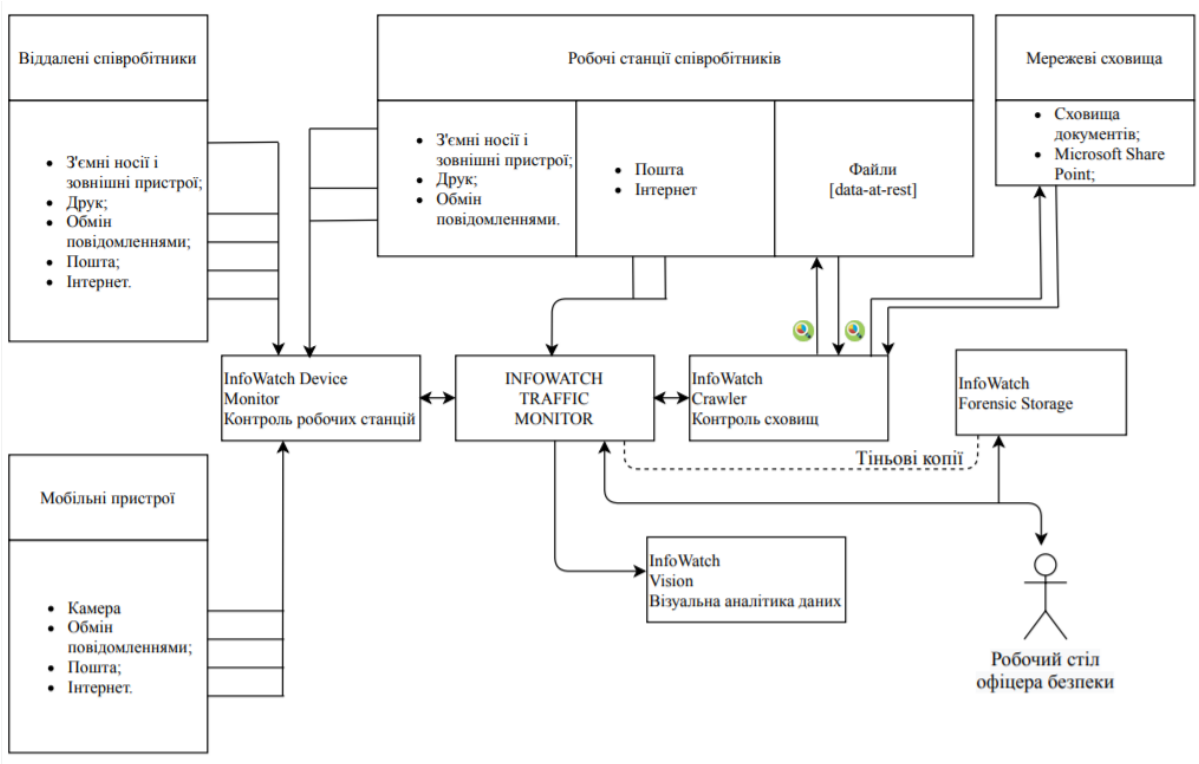


Рис. 1.3. Архітектура InfoWatch Traffic Monitor 6.7

Переваги:

- готовність реалізовувати функціонал, якщо потрібно;
- простота в установці (kickstart);
- масштабованість;
- багато бачить, багато розуміє — великий вибір каналів, робота з мобільним трафіком, великий вибір методів детектування конфіденційних даних;
- відносно швидкий пошук і зручний перегляд з підсвіткою різних об'єктів різними кольорами;

- докладний опис технологій і об'єктів захисту;
- автоматичне визначення критичності, розвантаження офіцера і т.п.;
- багато уваги приділяється візуалізації зібраної інформації. Одне з кращих рішень на ринку.

Недоліки:

- незв'язність консолей;
- нефункціональне розмежування доступу;
- один з модулів взагалі збирає інформацію, яку ядро аналізувати не може;
- використовується аж два агента, деякі функції включені в обидва;
- дані потрібно переганяти між базами;
- в один сервер повний функціонал помістити неможливо навіть в самому мінімальному впровадженні;
- логіка роботи Person Monitor (оперування не подія, а звітами) не дозволяє комфортно використовувати його для всієї зони покриття, а виключно «точково».

Рішення InfoWatch забезпечують ефективний контроль і аудит стану інфраструктури внутрішньої IT-безпеки організації. Завдяки багаторівневому моніторингу дій користувачів InfoWatch дозволяє створити комплексний захист конфіденційної інформації проти навмисних і необерних дій персоналу. Реалізація такої стратегії допомагає протистояти промисловому шпигунству та внутрішньому саботажу, мінімізувати операційні ризики, пов'язані із втратою конфіденційності даних.

Symantec DLP

Symantec DLP від компанії Symantec Corporation вирішує завдання з моніторингу, виявлення та захисту будь-якої корпоративної інформації в

будь-який час в будь-якому місці. Продукт може знаходити дані в хмарі, в локальних середовищах, в стаціонарних комп'ютерах і портативних пристроях.

Архітектуру системи Symantec DLP, можна переглянути на Рис. 1.3., що наведено нижче.

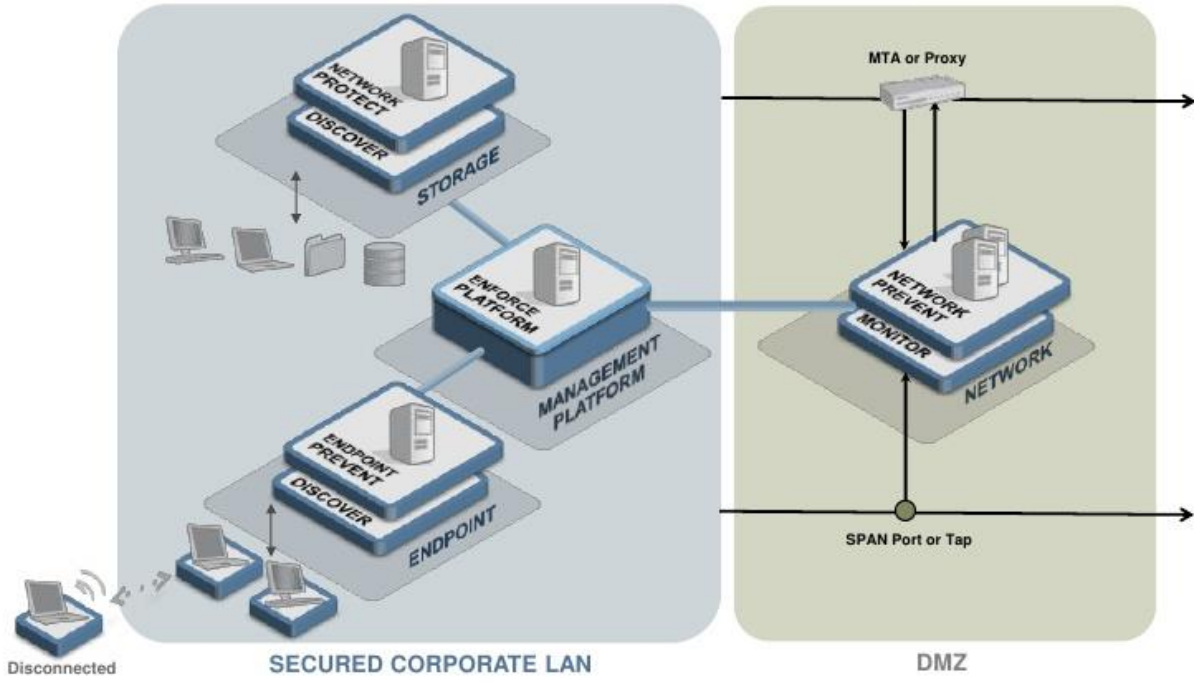


Рис. 1.4. Архітектура системи Symantec DLP

Далі розглядаються особливості системи, її переваги та недоліки. Що надасть змогу порівняти існуючі системи, та визначити ключові особливості програм створюваних у аспекті захисту конфеденційності.

Переваги:

- Велика кількість методів для аналізу: контентний, цифрові відбитки, автоматичне навчання, аналіз контексту, гібридний аналіз.
- Багаторівневий захист інфраструктури: функціональний агент, високопродуктивні мережеві компоненти, захист багатьох сховищ.

- Можливість масштабування системи для забезпечення функціонування в складних високонавантажених інфраструктура.
- Підтримка великої кількості мережевих протоколів для перехоплення і аналізу даних.
- Широкі можливості інтеграції як з лінійкою продуктів Symantec, так і зі сторонніми рішеннями.
- Endpoint-агент з можливістю блокування даних по всім заявленим каналам із здійсненням контентного аналізу.
- Модуль Data Insight (аналіз прав доступу на сховищах, аналіз доступу на сховищах, старіння даних і т.д.).
- Широкі можливості контролю HTTP / HTTPs трафіку, включаючи веб-пошту, соціальні мережі та інші довільні веб-сервіси (система записує повідомлення як HTTP (S) -трафік з певного сайту).
- Контроль мобільних пристроїв під управлінням Android і iOS.

Недоліки:

- Відсутність можливості контролю IM-протоколів таких як OSCAR, Mail.Ru Agent, Jabber, Microsoft Lync (компенсується контролем буфера обміну і контролем звернення додатків до конфіденційних файлів).
- Відсутність обробки на агентах індексів табличних даних.

Спеціалізований структурний підрозділ Державного центру кіберзахисту та протидії кіберзагрозам CERT-UA розробив рекомендації щодо захисту інформаційних ресурсів від внутрішніх загроз. Так, необхідно будувати систему захисту, орієнтуючись на попередній досвід компанії щодо внутрішніх інцидентів ІБ. Перш за все, необхідно забезпечити надійний захист критично важливих ресурсів, для цього необхідно використовувати різноманітні технології та системи захисту (DLP, SIEM, IDS тощо). Однак

жоден з них не може забезпечити повний захист від діяльності інсайдерів. Найважливішою складовою системи захисту конфіденційної інформації від витоків слід вважати наявність якісної політики безпеки [7].

Об'єктом дипломного проекту є створення системи моніторингу за витоком конфіденційної інформації, що дає змогу виявити наслідки атаки, що вже відбулася.

ВИСНОВОК ДО РОЗДІЛУ 1

У цьому розділі приведена загальна інформація про предмет інформаційної безпеки. Ключовими принципами якої є:

- Конфіденційність;
- Цілісність;
- Доступність;

Основними задачами систем інформаційної безпеки є:

- забезпечення захищеного зберігання інформації на носіях;
- захист даних, що передаються по каналах зв'язку;
- створення резервних копій, післяаварійного відновлення і т. д.

Каналами витоку інформації може служити:

- Людина;
- Апаратура;
- Програма;

Причинами утровення каналів витоку інформації можуть стати:

- втрати документів або конфіденційних записів;
- незнання або ігнорування персоналу фірми вимог щодо захисту інформації;

- зайва балакучість співробітників з колегами по роботі, іншими особами в місцях загального користування, у транспорті й т. д.;
- роботи з конфіденційними документами при сторонніх особах за рахунок несанкціонованої передачі їх іншому співробітникові;
- у результаті наявності в документах зайвої конфіденційної інформації;
- у результаті самовільного копіювання співробітником документів у службових або колекційних цілях.

Для запобігання витоку інформації найросповсюдженішим рішенням вважаються DLP-системи. Яскравими прикладами подібних систем є InfoWatch Traffic Monitor та Symantec DLP. Для їх опису приведені переваги і недоліки кожної з систем. В процесі вивчення предмету інформаційної безпеки було встановлено, що жодна з систем або їх комбінацій не дає повного захисту від втрати цінної інформації. За об’єкт дипломного проекту взято часткове рішеня для моніторингу і виявлення витоків конфіденційної інформації.

РОЗДІЛ 2
АНАЛІЗ СТАНУ СУЧАСНИХ ТЕХНОЛОГІЙ ДЛЯ СИСТЕМ МОНІТОРИНГУ І ВИЯВЛЕННЯ ВИТОКІВ ІНФОРМАЦІЇ

Технології інтегровані у існуючі системи захисту інформації.

Вище згадані продукти використовують різні інструменти детектування та моніторингу дій інсайдерів. Діяльність наведених технологій базується на технології детектування конфіденційної інформації [8]. У цьому розділі розглянуто найбільш розвинуті та розповсюджені технології детектування.

Пропонується оглянути список дій для забезпечення безпеки даних у організації з боку співробітників, це надасть можливість у подальшому чітко визначити потреби до створюваного продукту у запланованих межах.

Розглянемо рекомендації для співробітників, для захисту даних на робочому місці. Якщо співробітник покидає робоче місце, знаходячись в центрі проекту, що включає ділову конфіденційну інформацію, він має виконати низку запобіжних заходів мір захисту: защитить данные компании от посетителей или других лиц, которые не имеют права просматривать эту информацию

- Заблокуйте комп’ютер;
- Змініть налаштування, та встановіть надійний пароль для свого облікового запису
- Після зустрічі переглянути матеріали та виконати їх очистку
- Після друку, копіювання забрати свої документи одразу
- Зберігати конфіденційні документи у належних місцях.

Сигнатури.

Розглянемо метод, що вважається найпростіший метод контролю – сигнатури. Представляє собою пошук у потоці даних певної послідовності символів, зазвичай представлена не словом, а довільним набором символів, наприклад є певною міткою. У контексті сигнатур існує поняття «стопвираз», що застосовують як заборонену послідовність символів. Якщо система налаштована тільки на одне слово, то результат її роботи — визначення 100 % збігу, тобто метод можна віднести до детерміністського. Однак частіше пошук певної послідовності символів все ж таки застосовують при аналізі тексту. В переважній більшості випадків сигнатурні системи налаштовані на пошук декількох слів і частоту зустрічальності термінів.

До переваг цього методу можна віднести простоту поповнення словника заборонених термінів і очевидність принципу роботи, а також те, що це найбільш надійний спосіб, якщо необхідно знайти відповідність слова або виразу на 100 %.

Недоліки стають очевидними після початку промислового використання такої технології при визначенні витоків і налаштуванні правил фільтрації. Більшість виробників DLP-систем працюють для західних ринків, а англійська мова дуже «сигнатурна» – форми слів найчастіше утворюються за допомогою прийменників без зміни самого слова. В російській мові, наприклад, все набагато складніше, тому що у ній є приставки, закінчення, суфікси. Реальне застосування цього методу вимагає наявності лінгвіста або команди лінгвістів як на етапі впровадження, так і в процесі експлуатації та оновлення бази. Безсумнівним недоліком є і те, що «сигнатури» нестійкі до примітивного кодування, наприклад, заміною символів на схожі за зображенням.[12]

«Цифрові відбитки»

Різного типу хешфункції зразків конфіденційних документів називають «digital fingerprints». Суть всіх методів одна й та сама. Загальний

сценарій дії такий: набирається база зразків конфіденційних документів. Принцип роботи DG досить проста і часто цим і приваблює: DLP/IPC-системі передається якийсь стандартний документ-шаблон, з нього створюється «цифровий відбиток» і записується в базу даних DF. Далі в правилах контентної фільтрації настроюється процентна відповідність шаблону з бази. Наприклад, якщо налаштувати 75 % відповідності «цифровому відбитку» договору поставки, то при контентній фільтрації DLP виявить практично всі договори цієї форми. Іноді, до цієї технології відносять і системи на зразок «антиплагіат», однак остання працює тільки з текстовою інформацією, водночас як технологія «цифрових відбитків», у залежності від реалізації, може працювати і різним медійним контентом і застосовуватися для захисту авторських прав і перешкоди випадкового або навмисного порушення законів і нормативів інформаційної безпеки. До переваг технології «цифрових відбитків» (Digital Fingerprints) можна віднести простоту додавання нових шаблонів, досить високий ступінь детектування і прозорість алгоритму технології для співробітників підрозділів по захисту інформації. Основним недоліком є те, що, незважаючи на всю простоту і фактичну відсутність лінгвістичних методів, необхідно постійно оновлювати базу даних «цифрових відбитків». На відміну від «сигнатур», що не вимагає постійного оновлення бази словами, він вимагає оновлення бази «цифрових відбитків». До недоліків «цифрових відбитків» можна віднести те, що практично від «розширення бази словами» підтримка DLP в ефективному стані переходить на «пошук та індексування нових і змінених файлів , що є більш складним завданням, навіть якщо це робиться DLP-системою напівавтоматично. Великі компанії, в яких з'являється до десятка тисяч нових і оновлених документів кожен робочий день тільки на серверних сховищах часто просто не в змозі відслідковувати все це в режимі реального часу, не кажучи вже про персональні комп'ютери і ноутбуки.

«Мітки».

Суть цього методу полягає у призначенні спеціальних «міток» всередині файлів, що містять конфіденційну інформацію. З одного боку, такий метод дає стабільні та максимально точні відомості для DLP-системи, з іншого – потрібно досить сильні зміни в інфраструктурі мережі. У лідерів DLP/IPC-ринку реалізація даного методу не зустрічається, тому розглядати її докладно не має особливого сенсу. Можна лише зауважити, що, незважаючи на явне достоїнство «міток» – якість детектування, є багато суттєвих недоліків: від необхідності значної перебудови інфраструктури всередині мережі до введення безлічі нових правил і форматів файлів для користувачів. Насправді, застосування такої технології перетворюється у впровадження спрощеної системи документообігу.

Регулярні вирази.

Пошук за регулярними виразами («масками») є також давно відомим способом детектування необхідного вмісту, однак в DLP він став застосовуватися відносно нещодавно. Часто цей метод називають «текстовими ідентифікаторами». Регулярні вирази дозволяють знаходити збіги за формою даних, у ньому не можна точно зазначити точне значення даних, на відміну від «сигнатур». До переваг технології регулярних виразів у першу чергу варто віднести те, що вони дозволяють детектувати специфічний для кожної організації тип наповнення, починаючи від кредитних карток і закінчуючи назвами схем обладнання, специфічних для кожної компанії. Крім того, форми основних конфіденційних даних змінюються вкрай рідко, тому їх підтримка практично не вимагатиме часових ресурсів. До недоліків регулярних виразів можна віднести їх обмежену сферу застосування в рамках DLP/IPC-систем, так як знайти за допомогою них можна тільки конфіденційну інформацію лише певної

форми. Регулярні вирази не ожуть застосовуватися незалежно від інших технологій, однак можуть ефективно доповнювати їх можливості.

Лінгвістичні методи.

Найбільш поширеним на сьогоднішній день методом аналізу в DLP/IPC-системах є лінгвістичний аналіз тексту. Він настільки популярний, що часто саме він у просторіччі іменується «тематичною фільтрацією», тобто несе на собі характеристику всього класу методів аналізу вмісту. Є технології, які використовують лише «стоп-вирази», що вводять тільки на рівні коріння, а сама система вже становить повний словник; є що базуються на розставленні ваг на терміни, що найчастіше зустрічаються в тексті. Є у лінгвістичних методах і свої відбитки, що базуються на статистиці; наприклад, береться документ, рахується п'ятдесят найбільш уживаних слів, потім вибирається з 10 найуживаніших з них у кожному абзаці. Такий «словник» є практично унікальною характеристикою тексту і дозволяє знаходити в «клони» значущі цитати. До заслуг лінгвістичних методів у DLP можна віднести те, що в морфології та інших лінгвістичних методах високий ступінь ефективності, порівняно з сигнатурами, при набагато менших трудовитратах на впровадження і підтримку. У випадку з використанням лінгвістичних методів детектування немає необхідності відстежувати появу нових документів і направляти їх на аналіз у IPC-систему, так як ефективність лінгвістичних методів визначення конфіденційної інформації не залежить від кількості конфіденційних документів, частоти їх появи і продуктивності системи фільтрації вмісту. Недоліки лінгвістичних методів також досить очевидні, п рший з них – залежність від мови – якщо організація представлена в декількох країнах, бази конфіденційних слів і виразів доведеться створювати окремо для кожної мови і країни, з огляду на всю специфіку.

Ручне детектування («Карантин»).

Ручна перевірка конфіденційної інформації іноді називається «Карантин». Будь-яка інформація, яка потрапляє під правила ручної перевірки, наприклад, у ній зустрічається слово «ключ», потрапляє у консоль фахівця інформаційної безпеки. Він по черзі вручну переглядає цю інформацію та приймає рішення про пропуск, блокування або затримку даних. Коли дані блокуються або затримуються, відправнику надсилається відповідне повідомлення. Безперечною перевагою такого методу можна вважати найбільшу ефективність. Однак, такий метод у реальному бізнесі можна застосовувати лише для обмеженого обсягу даних, тому що потрібно велика кількість людських ресурсів, так як для якісного аналізу всієї інформації, що виходить за межі компанії, кількість співробітників інформаційної безпеки має приблизно збігатися з кількістю інших офісних співробітників. Реальне застосування для такого методу – аналіз даних обраних співробітників, де потрібна більш тонка робота, ніж автоматичний пошук за шаблонами, «цифрових відбитків» або збігів зі словами з бази.

Міжмережевий екран, мережевий екран - програмний або програмно-апаратний елемент комп'ютерної мережі, що здійснює контроль і фільтрацію проходить через нього мережевого трафіку відповідно до заданих правил. Серед завдань, які вирішують міжмережеві екрани, основний є захист сегментів мережі або окремих хостів від несанкціонованого доступу з використанням вразливих місць в протоколах мережевий моделі OSI або в програмному забезпеченні, встановленому на комп'ютерах мережі. Міжмережеві екрани пропускають або забороняють трафік, порівнюючи його характеристики з заданими шаблонами. Найбільш поширене місце для установки міжмережевих екранів - межа периметра локальної мережі для захисту внутрішніх хостів від атак ззовні. Однак атаки можуть починатися і з внутрішніх вузлів - в цьому випадку, якщо атакується хост розташований в тій же мережі, трафік не перетне кордон мережевого периметра, і міжмережевий екран не буде задіяний.

Honeytokens — це вигадані слова чи записи, які додаються до законних баз даних. Вони дозволяють адміністраторам відслідковувати дані в ситуаціях, які вони зазвичай не могли б відстежувати, наприклад, хмарні мережі. Якщо дані вкрадені, honeytokens дозволяють адміністраторам визначити, з кого вона була викрадена або як вона просочилася. Якщо для медичних записів є три місця, до кожної локації можна додати різні медові жетони у вигляді підроблених медичних записів. У кожному наборі записів були різні медотоки. Якщо вони обрані унікальними і навряд чи з'являться в законному трафіку, вони також можуть бути виявлені по мережі системою виявлення вторгнень (IDS), попереджаючи системного адміністратора про речі, які в іншому випадку залишаться непоміченими. Це один випадок, коли вони виходять за рамки просто забезпечення цілісності, а за допомогою деяких реактивних механізмів захисту можуть фактично запобігти шкідливій діяльності, наприклад. відкинувши всі пакети, що містять медосцену на маршрутизаторі. Однак у таких механізмів є підводні камені, оскільки це може спричинити серйозні проблеми, якщо honeypot був погано обраний і з'явився в законному мережевому трафіку, який потім був скинутий.

Технології розробки власної системи

Перш ніж почати основні розробки програмного забезпечення, ми повинні вибрати відповідну архітектуру, яка надасть нам бажану функціональність та якість атрибутів. Отже, ми повинні розуміти різні архітектури, перш ніж застосовувати їх до нашого дизайну.

Багаторівнева модель архітектури

Може бути використана для структури програм, які можна розкласти на групи підзадач, кожна з яких знаходиться на певному рівні абстракції. Кожен шар надає послуги наступному вищому шару. Найчастіше зустрічаються чотири шари загальної інформаційної системи:

- Презентаційний рівень (також відомий як рівень UI);
- Прикладний рівень (також відомий як службовий рівень);
- Рівень бізнес-логіки (також відомий як доменний рівень);
- Рівень доступу до даних (також відомий як постійний рівень).

Використання:

- Загальні настільні програми;
- Веб-програми електронної комерції.

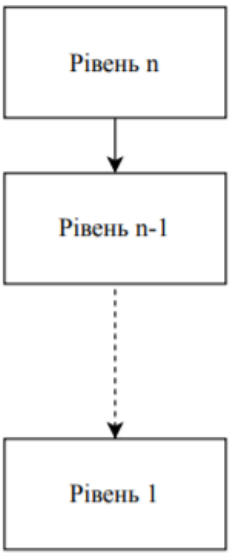


Рис. 2.1. Багаторівнева модель

Модель архітектури клієнт-сервер.

Архітектура клієнт-сервер означає наявність двох сторін: сервера і клієнта, яких може бути і декілька. Серверний компонент зосереджено на наданні послуг для кількох клієнтських компонентів, що є обробкою інформації згідно запита клієнта та відповідь на нього. А клієнти запитують послуги з сервера, що є надсиланням запиту, вигляд якого обумовлений стандартами, а сервер надає відповідні послуги цим клієнтам. Крім того, сервер продовжує слухати запити клієнтів. Застосовується у веб-додатках,

наприклад як електронна пошта, обмін документами та банківська діяльність.

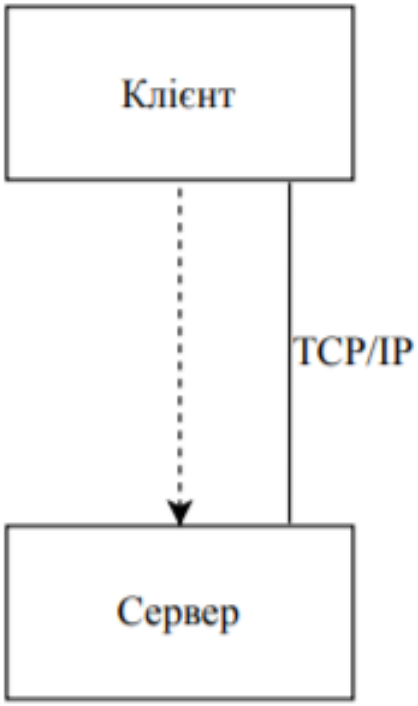


Рис. 2.2. Модель клієнт-сервер

Майстер-слуга модель архітектури

Ця модель складається з двох сторін; майстер і слуги. Головний компонент розподіляє роботу між однаковими компонентами слуг і обчислює кінцевий результат з результатів, які повертають слуги.

Використання:

- У реплікації бази даних головна база даних розглядається як авторитетне джерело, і підпорядковані бази даних синхронізуються з нею.
- Периферійні пристрої, підключені до шини в комп'ютерній системі (головні та підпорядковані пристрої).

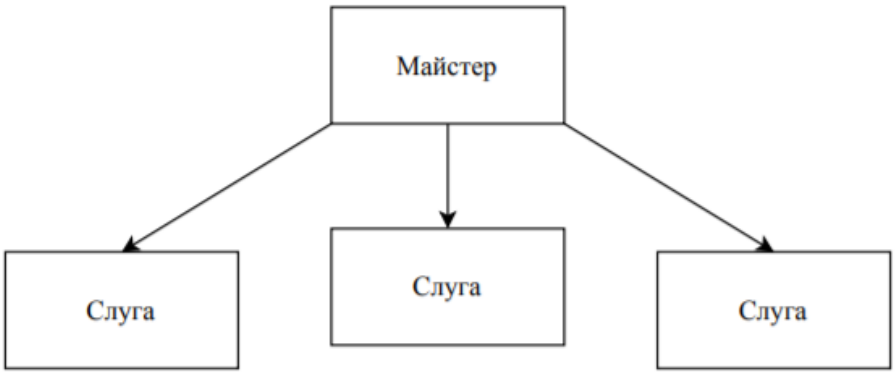


Рис. 2.3. Модель Майстер-слуга

Однорангова модель архітектури

У цій моделі окремі компоненти відомі як однолітки. Колеги можуть функціонувати і як клієнт, вимагаючи послуги від інших однолітків, і як сервер, надаючи послуги іншим одноранкам. Ровесник може виступати як клієнт, або як сервер, або як обидва, і він може з часом динамічно змінювати свою роль.

Використання:

- Мережі спільного використання файлів, такі як Gnutella та G2)
- Мультимедійні протоколи, такі як P2PTV та PDTP.

Рис. 2.4. Однорангова модель

Для даного проекту вирішено обрати клієнт-серверну архітектуру. За основу розробки системи обрали мову програмування Python. Python — інтерпретована мультипарадигменна мова програмування, призначена для розв'язання широкого спектру задач [9]. Основними перевагами використання Python у контексті даного проекту є:

- Мультиплатформеність. Оскільки інтерпретатор Python встановлений на усі популярні операційні системи, розроблену систему можна використовувати на будь-якій з них.
- Велика кількість користувацьких бібліотек. Python -- одна з найпопулярніших мов програмування, тому для Python створена велика кількість користувацьких бібліотек.
- Швидкість розробки. Декілька особливостей мови програмування Python дозволяють розробляти програмне забезпечення ефективно: гнучка динамічна система типів, підтримка функцій у якості об'єктів першого порядку (що дозволяє використовувати техніки функціонального програмування з метою оптимізації процесу

розробки по часу), підтримка міжпроцесної комунікації стандартною бібліотекою.

З цих причин для реалізації серверної частини системи та користувацького інтерфейсу буде використовуватись саме Python, тому доцільно розглянути компоненти та бібліотеки, що будуть використовуватись для реалізації основного функціоналу.

Для створення HTTP-серверу, що взаємодіє з клієнтською частиною системи був обраний мікрофреймворк для Python Flask. Flask призначений для створення веб-серверів та базується на Werkzeug, WSGI бібліотеці для Python [10]. В цілому для Python існує багато веб-фреймворків, але Flask обрали завдяки багатьом причинам описаним нижче:

- Змога працювати у контексті ізольованого процесу;
- Відсутність необхідності попередньої обробки для запуску проекту;
- Проста структура проекту завдяки використанню Flask;
- Змога використовувати вбудований веб-сервер, без CGI взаємодії з серверним програмним забезпеченням.

Оскільки робота системи повинна забезпечувати пряму взаємодію користувача з серверами з метою, наприклад, надсилання команд клієнтській частині системи, необхідно забезпечити комунікацію між компонентами користувацького інтерфейсу та серверів. У рамках даного проекту використовується стандартний модуль multiprocessing, який використовує процеси для забезпечення паралельного виконання замість потоків. Це рішення було прийняте з наступних причин:

- Використання процесів дозволяє ізолювати потоки різних серверів у окремих контекстах, оскільки компоненти для розробки серверів потребують власного керування потоками, а отже, мають власний event loop;

- Multiprocessing підтримує прозору систему комунікації за допомогою проксі-словників, що дозволяє швидку інтеграцію до Python-проектів;
- Інтерфейс multiprocessing дозволяє використовувати Python-функції у якості тіла процесу, що спрощує розробку серверних компонентів системи.

Для забезпечення цього функціоналу у рамках проекту була створена система гнучкої конфігурації процесу створення додатків за допомогою стандартного функціоналу Python. У якості компонентів для її розробки використовувались:

- Здатність Python з динамічного форматування строкових об’єктів з метою конфігурації додатків та їх використання під час створення експортованих файлів
- Здастність Python до виконання команд операційної системи з метою інтеграції системи з будь-якими інструментами, що підтримують інтерфейс командного рядка
- Здатність Python з взаємодії з файловою системою з метою створення файлів-агентів та взаємодії з шаблонами.
- YAML у якості зручної у прочитанні мови створення конфігураційних файлів з підтримкою об’єктів довільної структури. [11]

Приклад використання YAML для описання об’єктів конфігурації наведено на наступному рисунку:

Рис. 2.5. Конфігураційний файл

У якості інтерфейсу взаємодії з користувачем був обраний командний рядок з наступних причин:

- Більшість взаємодії з додатками потребує лише текстової інформації;
- Швидкість розробки, спричинена відносною простотою додавання команди у порівнянні зі створенням графічного інтерфейсу для нового функціоналу.

Для реалізації користувацького інтерфейсу у рамках проекту було прийняте рішення використовувати стандартний функціонал мови Python для спрощення взаємодії з іншими компонентами серверної частини

системи. Користувацький інтерфейс виконується у основному процесі додатку у вигляді циклу запиту команд:

```
while True:
 cmd_str = input('cmd:>')
 args = None
 if cmd_str == '':
  continue
 if " " in cmd_str:
  cmd, args = cmd_str.split(" ", 1)
 else:
  cmd = cmd_str
 if args:
  handlers[cmd](args)
 else:
  handlers[cmd]()
```

Рис. 2.6. Цикл розпізнавання команд користувача

ВИСНОВОК ДО РОЗДІЛУ 2

У другому розділі представленні технології, що застосовуються у сучасних складних системах спостереження та виявлення витоку інформації. А також огляд моделей організації архітектури програм. Обрано клієнт-сервер модель за основу дипломного проекту.

Проведено аналіз технологій для розробки клієнтської частини системи моніторингу і виявлення витоків конфіденційної інформації. Крім того, виконаний аналіз технологій та підходів для створення серверної частини додатку та були обрані наступні технології, мови програмування та модулі:

- Python — у якості мови програмування для створення серверів та користувацького інтерфейсу.
- Python, PowerShell — у якості мов програмування для взаємодії з шаблонами файлів-агентів.
- Flask— у якості бібліотеки для реалізації серверу.
- YAML —мова для реалізації об’єктів конфігурації.

Система працюватиме з файлами-агентами, які будуть виконувати роль HoneyToken.

РОЗДІЛ 3
РОЗРОБКА КІНЦЕВОГО ПРОДУКТУ

Загальний опис проекту.

Згідно поставлених задач дипломний проект поділяється на 3 складові:

- HTTP сервер;
- Користувацький інтерфейс;
- Шаблони для файлів-агентів та конфігураційні файли для реалізації клієнтської частини.

Усі компоненти проекту (HTTP сервер, інтерфейс користувача та генерація файлів-агентів) реалізовано з допомогою високорівневої мови програмування Python у межах однієї Python-програми. Файли-агенти для виявлення каналів витоку конфіденційної інформації формувалися за допомогою мови розмітки HTML та мови сценаріїв PowerShell. Файли конфігурації, що містять правила, за якими сервер визначає наскільки безпечним стався доступ до інформації, створені за допомогою YAML.

Структура проекту приведена на малюнку нижче, проект складається з програми, що піднімає сервер і спілкується з користувачем, шаблонів для файлів-агентів та конфігураційного файлу:

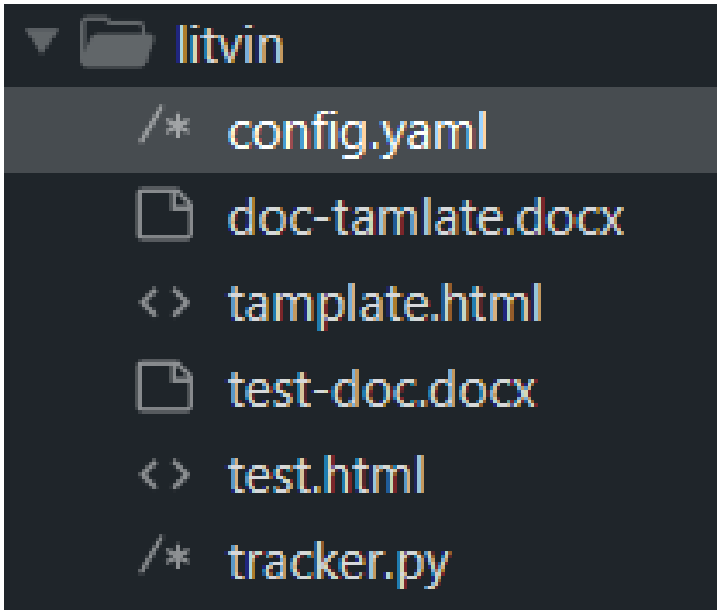


Рис. 3.1. Файли проекту

Сервер під час запуску обробляє об’єкт конфігурації. Тип цього об’єкту може різнитися, оскільки він формується під час синтаксичного аналізу конфігураційних файлів YAML. Приклад конфігурації серверу у вигляді витягу з конфігураційного файлу додатку показаний на наступному рисунку:

```yaml
server:
  host: "0.0.0.0"
  hostname: "185.130.55.87"
  port: 80
```

Рис. 3.2. Конфігурація HTTP сервера

Для першого кроку у роботі системи користувач створює унікальний код ідентифікатор файлу-агента. Сервер робить запис у словнику, де будуть зберігатися усі повідомлення щодо спроб доступитися до інформації у файлі-агенті. Користувач може коли завгодно зробити запит на аналіз результатів

спостереження. Якщо клієнт більше не зацікавлений у спостереженні за даним файлом-агентом, він може припинити процес. Для опису взаємодії додатку і користувача, створено UML-діаграма роботи клієнтської частини, що наведено на Рис. 3.3. нижче.

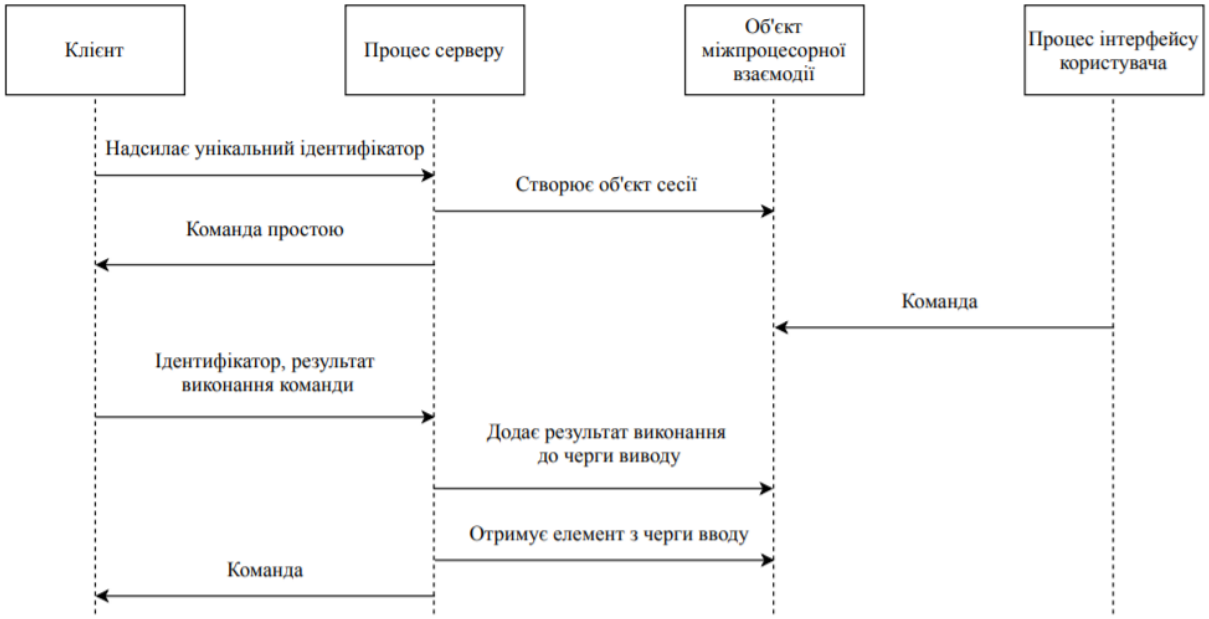


Рис. 3.3 UML-діаграма роботи клієнтської частини

До того ж, система налаштована на періодичне сканування результатів спостереження. Початкові дані методу автоматичного сканування результатів також зазначені у конфігураційному файлі.

Клієнт також здатен налаштувати групи і правила за якими члени даної групи можуть отримати санкціонований доступ до інформації. Кожна група має свою назву, тип файлу і правила доступу до файлів за даним типом. Все це виглядає наступним чином:

Рис. 3.4. Приклад розмежування прав доступу

У випадку, якщо система автоматичного сканування виявила несанкціонований доступ до інформації, вона може автоматично надіслати повідомлення користувачу.

Більшість переваг використання архітектури клієнт-сервер для програм стосується гнучкості розгортання та відносної простоти обслуговування. Наприклад, використовуючи архітектуру клієнт / сервер, ти зазвичай:

- Повторне використання існуючого застарілого коду для бізнес-логіки;
- Запуск кожного функціонального шару програми на платформі, що найбільш підходить до завдання;
- Розподіл обробки та мережевих навантажень;
- Швидка та легка зміна процедури ділової логіки без зміни клієнтської програми чи інтерфейсу користувача;
- Забезпечення простої та зручної доставки програми та будь-яких оновленнь для кінцевих користувачів

- Забезпечення альтернативних інтерфейсів клієнтів для тієї самої програми на стороні сервера;
- Використання засобів розробки, розроблених для спільної роботи;
- Щоб максимально використати потенційну цінність архітектури клієнт-сервер, слід дотримуватися деяких основних рекомендацій щодо дизайну. Вони викладені нижче.

Набір функціоналу, доступного для користувача

Для скористання програмою користувачу необхідно встановити необхідні бібліотеки Python. Для наглядного прикладу нижче на Рис. 3.5. наведено встановлення деяких необхідних пакетів, за допомогою утиліти pip. Вона забезпечує пошук відповідного пакета у інтернеті, заантаження його на пристрій, розархівування та налаштування. На Рис. 3.5. зображено налаштування пакетів Flask, datetime та schedule.

```
C:\Users\bonanza>pip install flask
Requirement already satisfied: flask in c:\users\bonanza\appdata\local\programs\
Requirement already satisfied: click>=5.1 in c:\users\bonanza\appdata\local\prog
Requirement already satisfied: Jinja2>=2.10.1 in c:\users\bonanza\appdata\local\
Requirement already satisfied: Werkzeug>=0.15 in c:\users\bonanza\appdata\local\
Requirement already satisfied: itsdangerous>=0.24 in c:\users\bonanza\appdata\lo
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\bonanza\appdata\loca

C:\Users\bonanza>pip install flask
Requirement already satisfied: flask in c:\users\bonanza\appdata\local\programs\
Requirement already satisfied: itsdangerous>=0.24 in c:\users\bonanza\appdata\lo
Requirement already satisfied: click>=5.1 in c:\users\bonanza\appdata\local\prog
Requirement already satisfied: Werkzeug>=0.15 in c:\users\bonanza\appdata\local\
Requirement already satisfied: Jinja2>=2.10.1 in c:\users\bonanza\appdata\local\
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\bonanza\appdata\loca

C:\Users\bonanza>pip install datetime
Requirement already satisfied: datetime in c:\users\bonanza\appdata\local\progra
Requirement already satisfied: pytz in c:\users\bonanza\appdata\local\programs\p
Requirement already satisfied: zope.interface in c:\users\bonanza\appdata\local\
Requirement already satisfied: setuptools in c:\users\bonanza\appdata\local\prog

C:\Users\bonanza>pip install schedule
Requirement already satisfied: schedule in c:\users\bonanza\appdata\local\progra
```

Рис. 3.5. Використання утиліти pip

В цьому випадку дуже зручно користуватися вбудованою утилітою pip, яка має чудове і надзвичайно просте представлення інтерфейсу у вигляді командної строки. Для встановлення необхідних пакетів достатньо ввести команду «pip install {назва пакету}».

Інтерфейс користувача має вигляд командного рядка. Користувачу доступні наступні команди:

- Створення нового ідентифікатора сесії для файла-агента;
- Створення агента у вигляді посилання;
- Створення файлу-агента у вигляді файлу HTML;
- Створення файлу-агента у вигляді doc-файлу;
- Запит на аналіз даних за ідентифікатором сесії;
- Створення нової групи;
- Додавання нового члена групи;
- Додавання нового правила до групи.

Базовий функціонал програми:

- help — вивід інформації про можливості користувача;
- info — виводить усю важливу інформацію про стан сесій і попередження;
- quit — припинення роботи програми.

Рис. 3.6. Можливості користувача

Створення файлів-агентів. На даному етапі розробки користувачу доступно створити файли-агенти типів HTML та Doc. Шаблон HTML-файлу можна переглянути на Рис. 3.7.

```
1  <!DOCTYPE html><html><head></head><body></body></html>
```

Рис. 3.7. шаблон HTML файлу

Простий приклад згенерованого HTML-файлу наведено на Рис. 3.8.

```
<!DOCTYPE html><html><head></head><body><img src="http://185.130.55.87/track?id=fGas3io7"/></body></html>
```

Рис. 3.8. Простий приклад згенерованого файлу

Коли до файлу будуть намагатися здійснити доступ, тобто відкрити його, він автоматично надішле запит на сервер.

Це дозволить серверу дізнатися час доступу до файлу, IP-адресу. На основі цієї інформації користувач має змогу формувати групи за часом доступу до файлу, IP-адресами або маскою підмережі. Якщо сервер помітить активність заборонену режимом доступу до документів, він викличе функція, що надішле господарю оповіщення.

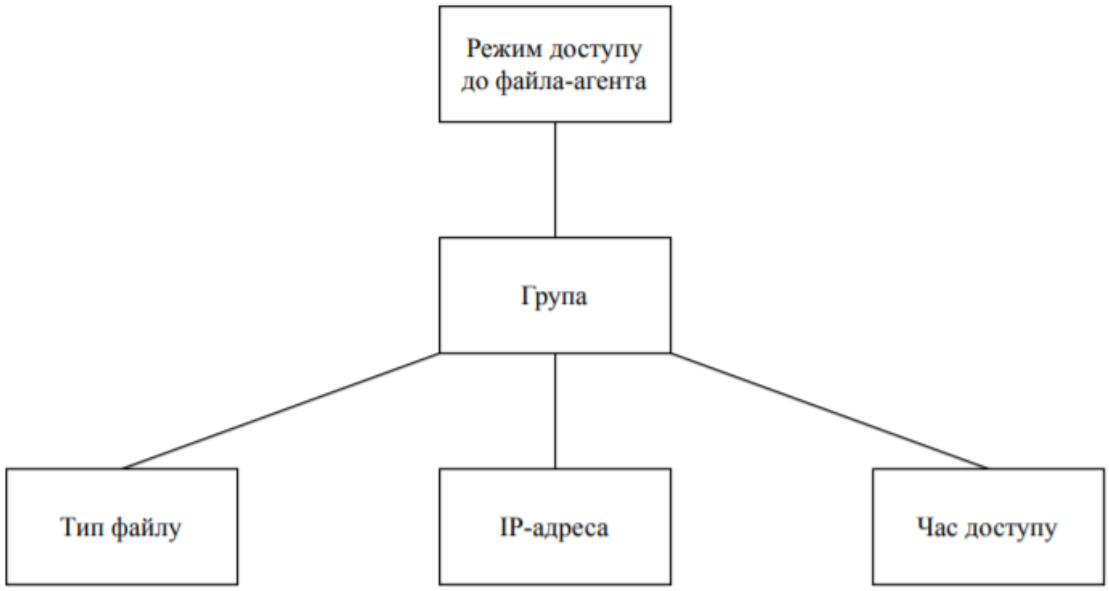


Рис. 3.9. Складові режиму доступу до файла

ВИСНОВОК ДО РОЗДІЛУ 3

В процесі розробки створено систему забезпечення політики безпеки доступу до документів. Що надає можливість користувачу спостерігати та виявляти несанкціонований доступ до документів HTML та doc форматів. Система реалізована мовою Python.

Система надає користувачу можливості:

- Створення файлів-агенті;
- Редагування груп та правил доступу до документів;
- Аналізувати стан сесій та отримувати повідомлення про несанкціонований доступ.

Сервер реалізовано за допомогою фреймворку Flask. Коли відбувається спроба отримати доступ до файлів, сервер отримує інформацію про ціль. Робить запис у журнал, аналізує дану подію відповідно до груп і правил користування файлом за його ідентифікатором. Після чого встановлює чи належить дана подія до запланованих або відноситься до факту витоку інформації. Після чого згідно налаштувань повідомляє про подію користувача.

Система включає в себе шаблони файлів-агентів, зручний вигляд користувацького інтерфейсу, а саме командного рядка. Конфігурація системи здійснюється за допомогою конфігураційного файлу у зручному представленні за допомогою мови YAML.

ЗАГАЛЬНІ ВИСНОВКИ

Питання безпеки інформації є складною і гострою проблемою. Створення різноманітних систем запезбечення безпеки інформації має допомогти задовольнити потреби різних сфер для безпечного користування комп'ютерними мережами і інформаційними технологіями.

Розроблена система забезпечення політики безпеки доступу до документів. Вона дозволяє шляхом використання файлів-агентів встановити групи і режими доступу файлів. У випадку несанкціонованого доступу виявити факт витоку інформації.

Під час аналізу нинішнього стану технологій захисту інформації були розглянуті такі методи, як: сигнатури, цифрові відбитки, мітки, лістингові методи, карантин, міжмережевий екран, honeytoken технологія.

У даній дипломній роботі було створено систему створення файлів-агентів. Система за ідентифікатором файлів здійснює постійне спостереження. На основі груп і правил встановлює режими доступу до файлів-агентів. У випадку неочікуваної події відбувається виявлення несанкціонованого доступу до документів за допомогою функції-аналізатора. Після чого система надсилає повідомлення користувачу.

У роботі вдалося досягнути мети створення системи спостереження і виявлення витоків інформації. Предмет дослідження було опрацьовано у повному обсязі, усі поставлені цілі дипломного проекту досягнуті.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Конфіденційна інформація [Електронний ресурс] — Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Конфіденційна_інформація
2. Інформаційна безпека [Електронний ресурс] — Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Інформаційна_безпека
3. Ульянов В.В. Динамика безопасности: от внешних угроз – к внутренним / В.В.Ульянов // Защита информации. INSIDE.– 2008. - № 4. – С. 34 – 38.
4. Probst, C.W. Insider Threats in Cyber Security/ C.W. Probst– 2010. – 245p.
5. 2006 CSI/FBI Computer Crime and Security Survey [Електронний ресурс] — Режим доступу до ресурсу: http://www.infowatch.ru/threats?chapter=147 151396&id=2926721
6. Cyber Cop Scanner [Електронний ресурс] — Режим доступу до ресурсу: http://www.nss.co.uk/grouptests/va/edition2/nai_cybercop_scanner/nai_cybercop_scanner.htm
7. CUA-15-04R. Рекомендації CERT-UA з протидії загрозі інсайдера [Електронний ресурс] / І. Соколов - №1 – 2015 — Режим доступу до ресурсу: http://cert.gov.ua/pdf/CUA-15-04R.pdf
8. World Wide Digital Security [Електронний ресурс] — Режим доступу до ресурсу: http://www.pcworld.com/article/id,143371-c,privacysecurity/article.html
9. A Python Book: Beginning Python, Advanced Python, and Python Exercises / Dave Kuhlman., 2012.
10. Flask Documentation [Електронний ресурс] — Режим доступу до ресурсу: https://flask.palletsprojects.com/en/1.1.x/.
11. YAML [Електронний ресурс] — Режим доступу до ресурсу: https://yaml.org/.
12. Технології захисту конфеденційної інформації від внутрішніх загроз, А.О. Антонюк, В.С. Портяной, В.П. Шилін/ С.W. Probst– 2011. – 88p.

13. Рис. 1.3. [Електронний ресурс] — Режим доступу до ресурсу https://www.open-vision.ru/catalog/security/dlp-system/symantec-data-loss-prevention/
14 Державні інформаційні ресурси. методологія побудови класифікатора загроз [Електронний ресурс] — Режим доступу до ресурсу: https://er.nau.edu.ua/bitstream/NAU/31911/1/Monogr_klas_zagroz_Yudin_Buchyk.pdf
15. Дані [Електронний ресурс] — Режим доступу до ресурсу http://perimetrix.ru/
16. Статистика втрати даних [Електронний ресурс] — Режим доступу до ресурсу https://comsecglobal.com/the-evolution-of-data-loss-prevention/

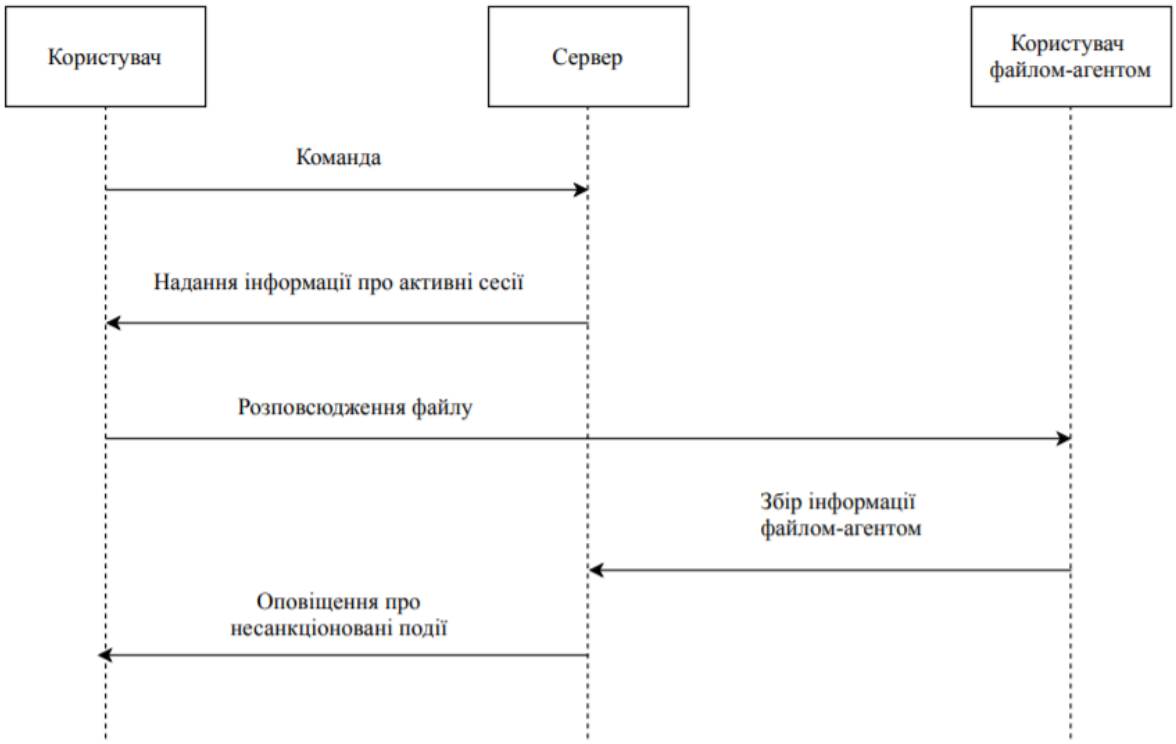
Додаток 1

Функціональна схема

до дипломного проекту

на тему: «Система забезпечення політики безпеки доступу до документів»

Київ – 2020 року



					ІАЛЦ. 467200.004 Д1			
Зм.	Арк.	№ докум.	Підпис	Дата				
Розробив		Літвін О.Ю,			Система забезпечення політики безпеки доступу до документів	Літ.	Аркуш	Аркушів
Перевір.							1	1
					Функціональна схема	НТУУ “КПІ ім. Ігоря Сікорського”, ФІОТ,ІО-61		
Н. контр.		Сімоненко В.П.						
Затверд.								

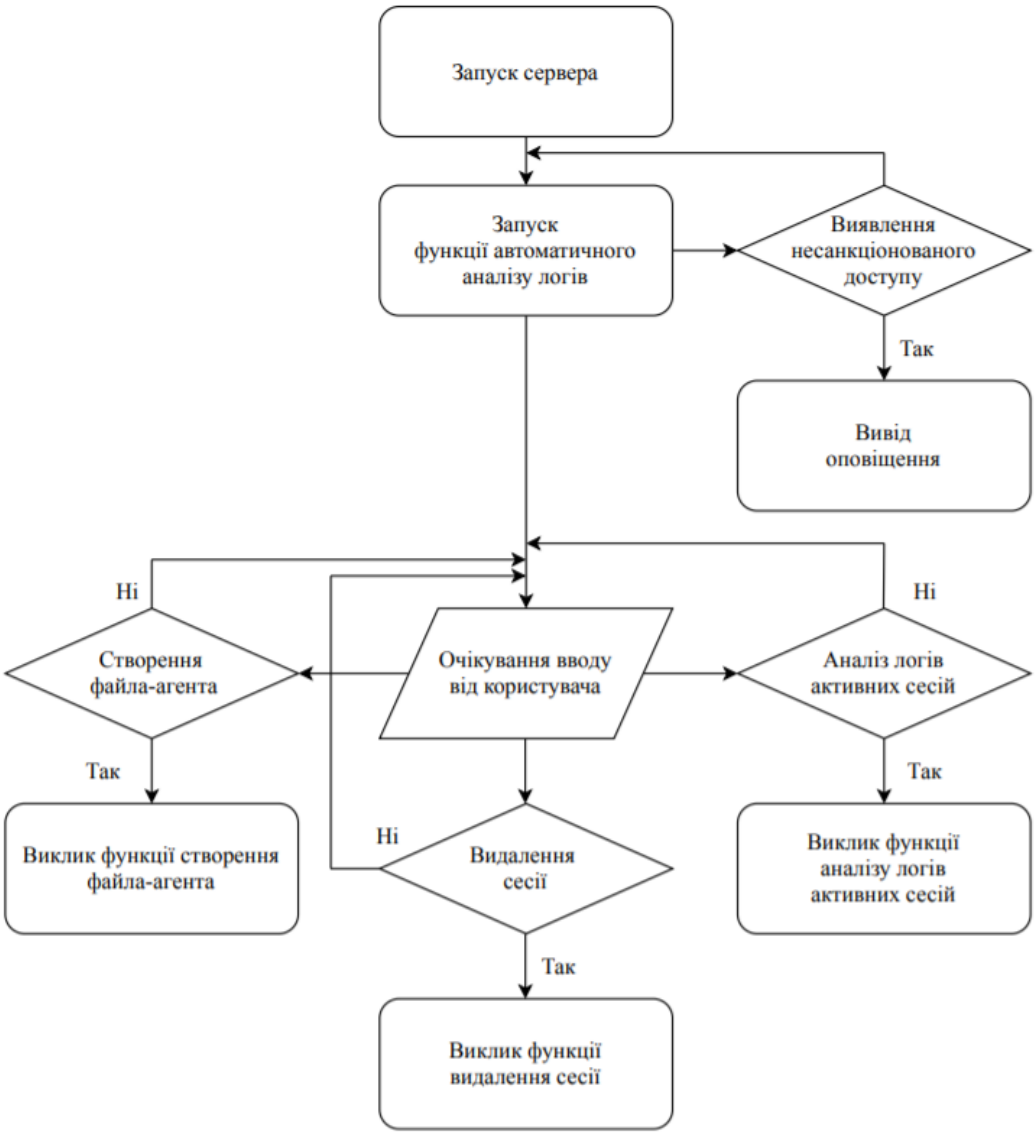
Додаток 2

Принципова схема

до дипломного проекту

на тему: «Система забезпечення політики безпеки доступу до документів»

Київ – 2020 року



					ІАЛЦ. 467200.005 Д2			
Зм.	*Арк.*	*№ докум.*	*Підпис*	*Дата*				
Розробив		*Літвін О.Ю.*			*Система забезпечення політики безпеки доступу до документів*	*Літ.*	*Аркуш*	*Аркушів*
Перевір.							1	1
					Принципова схема	*НТУУ "КПІ ім. Ігоря Сікорського", ФІОТ, ІО-61*		
Н. контр.		*Сімоненко В.П.*						
Затверд.								

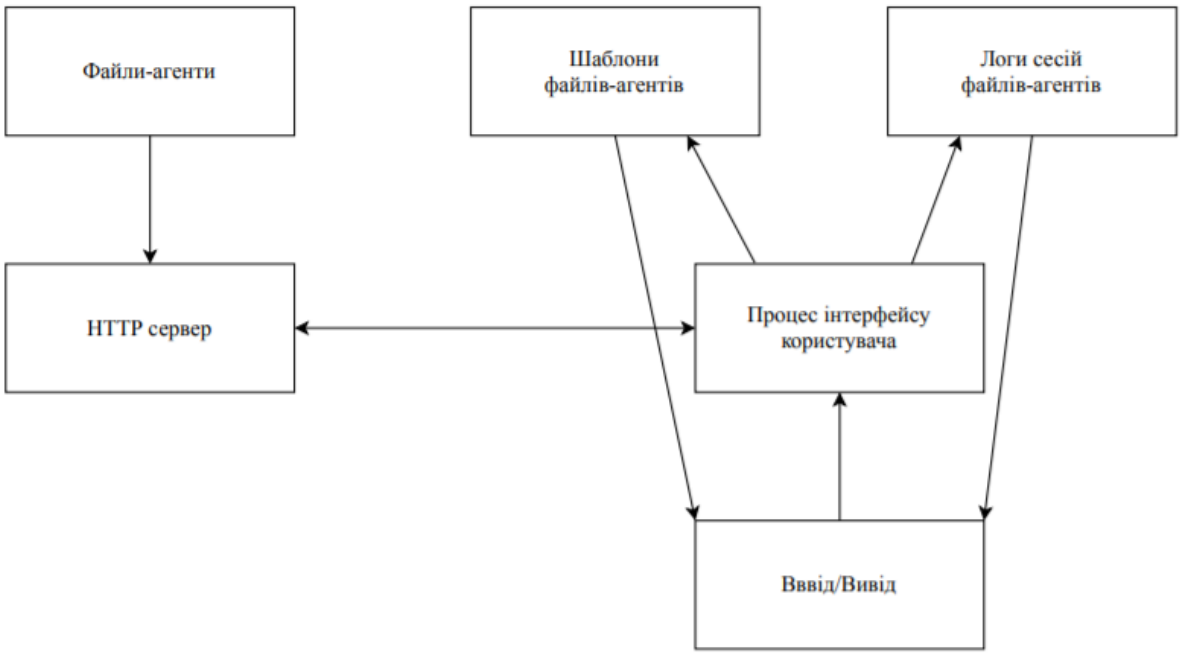
Додаток 3

Структурна схема

до дипломного проекту

на тему: «Система забезпечення політики безпеки доступу до документів»

Київ – 2020 року



					ІАЛЦ. 467800.006 ДЗ		
Зм.	Арк.	№ докум.	Підпис	Дата			
Розробив		Літвін О.Ю.			Система забезпечення політики безпеки доступу до документів	Літ.	Аркуш 1 / Аркушів 1
Перевір.					Структурна схема		
Н. контр.		Сімоненко В.П.				НТУУ “КПІ ім. Ігоря Сікорського”, ФІОТ, ІО-61	
Затверд.							

Додаток 4

Лістинг програми

до дипломного проекту

на тему: «Система забезпечення політики безпеки доступу до документів»

Київ – 2020 року

tracker.py

```
import logging, sys
import schedule, time
from datetime import datetime



print("[+] loading configuration")
from yaml import safe_load
with open("config.yaml", 'r') as f:
 config = yaml.safe_load(f)


def run_tracking(share):
 print("[+] starting HTTP tracking listener...")
 from flask import Flask, request


 log = logging.getLogger('werkzeug')
 log.setLevel(logging.ERROR)
 cli = sys.modules['flask.cli']
 cli.show_server_banner = lambda *x: None


 app = Flask("tracking_server")

 @app.route("/track")
 def track_req():
  print("[+] new tracking request from {} with id:{}".format(request.remote_addr,
request.args.get("id")))
  share["tracks"] += [{"ip": request.remote_addr, "id": request.args.get("id"), "time":
datetime.now().strftime("%d-%b-%Y (%H:%M:%S.%f)")}]
  return '', 200

 app.run(port=share["config"]["server"]["port"],
host=share["config"]["server"]["host"])

def scheduler():
  schedule.every(10).minutes.do(analyze)
  time.sleep(1)
```

```
import multiprocessing

manager = multiprocessing.Manager()
share = manager.dict()
share["config"] = config
share["tracks"] = []
share["ids"] = []
http_server = multiprocessing.Process(target=run_tracking, args=(share,))
http_server.start()



http_host = config["server"]["hostname"]




cmd = "

handlers = {}

def quit():
 http_server.terminate()
 sys.exit(0)
handlers["quit"] = quit

import string, random
def random_string(len):
    letters = string.ascii_lowercase + string.ascii_uppercase + string.digits
    return ''.join(random.choice(letters) for i in range(len))



def newid(args):
 typeid, group = args.split(" ")
 new_id = random_string(8)
 share["ids"] += [{"id": new_id, "type": typeid, "group":group}]
 print("id for type {} will be {}".format(args, new_id))
handlers["newid"] = newid

def newrule(args):
 groupname, typename, permission, source, accesstime = args.split(" ")
```

```
 with open(config.yaml, 'r') as f:
   file_c = f.read()
  file_c = file_c.replace("#newrule:", """{}:
      {}:
      rules:
        -action:{}
         source:{}
         accestime:{}")""".format(groupname, typename, permission, source,
accesstime)
handlers["rule"] = newrule

def ids():
 for i in share["ids"]:
  print(i)
handlers["ids"] = ids


def export_as_link(args):
 id_obj = None
 for i in share["ids"]:
  if i["type"] == args:
   id_obj = i
 img_template = "<img src=\"http://{}/track?id={}\"/>"
 if id_obj:
  print(img_template.format(http_host, id_obj["id"]))
handlers["link"] = export_as_link

def export_to_doc(args):
 type_str, filename = args.split(" ", 1)
 id_obj = None
 for i in share["ids"]:
  if i["type"] == type_str:
   id_obj = i
 if id_obj:
  img_template = "<img src=\"http://{}/track?id={}\"/>".format(http_host,
id_obj["id"])
  subprocess.call(["export_to_doc.ps1"])
handlers["file"] = export_to_file

def export_to_file(args):
 type_str, filename = args.split(" ", 1)
 id_obj = None
```

```
 for i in share["ids"]:
  if i["type"] == type_str:
   id_obj = i
 if id_obj:
  img_template = "<img src=\"http://{}/track?id={}\"/>".format(http_host,
id_obj["id"])
  with open(filename, 'r') as f:
   file_c = f.read()
  file_c = file_c.replace("</body>", img_template+"</body>")
  with open(filename, 'w') as f:
   f.write(file_c)
handlers["file"] = export_to_file

def analyze():
 for event in share["tracks"]:
  id_obj = None
  for id_config in share["ids"]:
   if id_config["id"] == event["id"]:
    id_obj = id_config

  if id_obj == None:
   print("[-] invalid access with id {}".format(event["id"]))
  else:
   acl = config["groups"][id_obj["group"]]
   for rule in acl["rules"]:
    if rule["source"] == "any" or event["ip"] in rule["source"]:
     if rule["action"] == "allow":
      print("allowed access from {}@{} at {}".format(event["ip"], event["id"],
event["time"]))
     else:
      print("[!] violation \ndisallowed access from {}@{} at {}".format(event["ip"],
event["id"], event["time"]))
     break

handlers["analyze"] = analyze




while True:
 cmd_str = input('cmd:>')
 args = None
```

```
 if cmd_str == '':
  continue
 if " " in cmd_str:
  cmd, args = cmd_str.split(" ", 1)
 else:
  cmd = cmd_str
 if args:
  handlers[cmd](args)
 else:
  handlers[cmd]()
```

config.yaml

```
server:
  host: "127.0.0.1"
  hostname: "185.130.55.87"
  port: 88

groups:
    default:
      type: html
      rules:
        - action: allow
          source: [127.0.0.1, 192.168.1.45]
        - action: deny
          source: any
    important:
      type: html
      rules:
        - action: deny
          source: any
    available:
      type: doc
      rules:
        -action: allow
         source: any
    limitedbytime:
      type: doc
        -action: allow
         accesstime: [(15:00,18:00), (23:00, 00;00)]
    #newrule:
```

export_to_doc.ps1

public Microsoft.Office.Interop.Word.InlineShape AddPicture (string "doc-tamplate", ref object ".format(http_host, id_obj["id"], ref object SaveWithDocument, ref object Range);